

Routing Open Shop with unit processing times, few machines, and few locations[☆]

René van Bevern^{a,b,1,*}, Artem V. Pyatkin^{a,b,2}

^aNovosibirsk State University, Novosibirsk, Russian Federation

^bSobolev Institute of Mathematics, Siberian Branch of the Russian Academy of Sciences, Novosibirsk, Russian Federation

Abstract

OPEN SHOP is a classical scheduling problem: given a set \mathcal{J} of jobs and a set \mathcal{M} of machines, find a minimum-makespan schedule to process each job $J_i \in \mathcal{J}$ on each machine $M_q \in \mathcal{M}$ for a given amount p_{iq} of time such that each machine processes only one job at a time and each job is processed by only one machine at a time. In ROUTING OPEN SHOP, the jobs are located in the vertices of an edge-weighted graph $\mathcal{G} = (V, E)$, whose edge weights determine the time needed for the machines to travel between jobs. The travel times also have a natural interpretation as sequence-dependent family or batch setup times. ROUTING OPEN SHOP is NP-hard for $|V| = |\mathcal{M}| = 2$. For the special case with unit processing times $p_{iq} = 1$, we exploit a variant of Galvin's theorem about list-coloring edges of bipartite graphs to prove a theorem that gives a sufficient condition for the completability of partial schedules. Exploiting this schedule completion theorem and integer linear programming, we show that ROUTING OPEN SHOP with unit processing times is solvable in $2^{O(|V||\mathcal{M}|^2 \log |V||\mathcal{M}|)} \cdot \text{poly}(|\mathcal{J}|)$ time, that is, fixed-parameter tractable parameterized by $|V| + |\mathcal{M}|$. Various upper bounds shown using the schedule completion theorem suggest it to be likewise beneficial for the development of approximation algorithms.

Keywords: combinatorial optimization, routing, scheduling, graph theory, fixed-parameter tractability

1. Introduction

One of the most fundamental and classical scheduling problems is OPEN SHOP (Gonzalez and Sahni, 1976), where the input is a set $\mathcal{J} := \{J_1, \dots, J_n\}$ of jobs, a set $\mathcal{M} := \{M_1, \dots, M_m\}$ of machines, and the processing time p_{iq} that job J_i needs on machine M_q ; the task is to process all jobs on all machines in a minimum amount of time such that each machine processes at most one job at a time and each job is processed by at most one machine at a time.

Averbakh et al. (2006) introduced the variant ROUTING OPEN SHOP, where the jobs are located in the vertices of an edge-weighted graph, whose edge weights determine the time needed for the machines to travel between jobs. Initially, the machines are located in a depot. The task is to minimize the time needed for processing all jobs by all machines and returning all machines to the depot. ROUTING OPEN SHOP models, for example, tasks where machines have to perform maintenance work on stationary objects in a workshop (Averbakh et al., 2006). ROUTING OPEN SHOP has also been interpreted as a variant of OPEN SHOP with sequence-dependent family or batch setup times (Allahverdi et al., 2008; Zhu and Wilhelm, 2006). Formally, ROUTING OPEN SHOP is defined as follows.

Definition 1.1 (ROUTING OPEN SHOP). An instance of ROUTING OPEN SHOP consists of a graph $\mathcal{G} = (V, E)$ with a depot $v^* \in V$ and travel times $c: E \rightarrow \mathbb{N} \setminus \{0\}$, jobs $\mathcal{J} = \{J_1, \dots, J_n\}$ with locations $\mathcal{L}: \mathcal{J} \rightarrow V$, machines $\mathcal{M} = \{M_1, \dots, M_m\}$, and, for each job J_i and machine M_q , a processing time $p_{iq} \in \mathbb{N}$.

A route with s stays is a sequence $R := (R_i)_{i=1}^s$ of stays $R_i = (a_i, v_i, b_i) \in \mathbb{N} \times V \times \mathbb{N}$ from time a_i to time b_i in the vertex v_i for $1 \leq i \leq s$ such that $v_1 = v_s = v^*$, $a_1 = 0$, and $b_i + c(v_i, v_{i+1}) \leq a_{i+1} \leq b_{i+1}$ for $1 \leq i \leq s-1$. The length of R is the end b_s of the last stay.

A schedule $S: \mathcal{J} \times \mathcal{M} \rightarrow \mathbb{N}$ is a total function determining the start time $S(J_i, M_q)$ of each job J_i on each machine M_q . That is, each job J_i is processed by each machine M_q in the half-open time interval $[S(J_i, M_q), S(J_i, M_q) + p_{iq})$. A schedule is feasible with respect to routes $(R_{M_q})_{M_q \in \mathcal{M}}$ if

- (i) no machine M_q processes two jobs $J_i \neq J_j$ at the same time, that is, $S(J_i, M_q) + p_{iq} \leq S(J_j, M_q)$ or $S(J_j, M_q) + p_{jq} \leq S(J_i, M_q)$ for all jobs $J_i \neq J_j$ and machines M_q ,
- (ii) no job J_i is processed by two machines M_q, M_r at the same time, that is, $S(J_i, M_q) + p_{iq} \leq S(J_i, M_r)$ or $S(J_i, M_r) + p_{ir} \leq S(J_i, M_q)$ for all jobs J_i and machines $M_q \neq M_r$,
- (iii) machines stay in the location $\mathcal{L}(J_i)$ while executing a job J_i , that is, for each job J_i and machine M_q with route $R_{M_q} = (R_k)_{k=1}^s$, there is a $k \in \{1, \dots, s\}$ such that $R_k = (a_k, \mathcal{L}(J_i), b_k)$ with $a_k \leq S(J_i, M_q) \leq S(J_i, M_q) + p_{iq} \leq b_k$.

A schedule S is feasible and has length L if there are machine routes $(R_{M_q})_{M_q \in \mathcal{M}}$ of length L such that S is feasible with respect to $(R_{M_q})_{M_q \in \mathcal{M}}$. An optimal solution to a ROUTING OPEN SHOP instance is a feasible schedule of minimum length.

[☆]An extended abstract of this article appeared at CSR'16 (van Bevern and Pyatkin, 2016). This version provides all proofs, additional figures, and a simpler but stronger version of the schedule completion theorem.

*Corresponding author

Email addresses: rvb@nsu.ru (René van Bevern), artem@math.nsc.ru (Artem V. Pyatkin)

¹Supported by the Russian Foundation for Basic Research (RFBR) under research project 16-31-60007 mol_a_dk.

²Supported by RFBR projects 15-01-00462 and 15-01-00976.

Preemption and unit processing times. OPEN SHOP is NP-hard for $|M| = 3$ machines (Gonzalez and Sahni, 1976). Thus, so is ROUTING OPEN SHOP with $|V| = 1$ vertex and $|M| = 3$ machines. ROUTING OPEN SHOP remains (weakly) NP-hard even for $|V| = |M| = 2$ (Averbakh et al., 2006); there are approximation algorithms both for this special and the general case (Averbakh et al., 2005; Chernykh et al., 2013; Kononov, 2015; Yu et al., 2011). However, OPEN SHOP is solvable in polynomial time if

- (1) job preemption is allowed, or
- (2) all jobs J_i have unit processing time $p_{iq} = 1$ on all machines M_q .

It is natural to ask how these results transfer to ROUTING OPEN SHOP.

Regarding (1), Pyatkin and Chernykh (2012) have shown that ROUTING OPEN SHOP with allowed preemption is solvable in polynomial time if $|V| = |M| = 2$, yet NP-hard for $|V| = 2$ and an unbounded number $|M|$ of machines.

Regarding (2), ROUTING OPEN SHOP with unit processing times models tasks where machines process batches of equal-length jobs in several locations (or of different types) and where the transportation of machines between the locations (or the setup between jobs of different types) takes significantly longer than processing each individual job in a batch. Herein, there are conceivable situations where the number of machines and locations are small.

ROUTING OPEN SHOP with unit processing times clearly is NP-hard even for $|M| = 1$ machine since it generalizes the metric travelling salesperson problem. It is not obvious whether it is solvable in polynomial time even when both $|V|$ and $|M|$ are fixed. We show the even stronger result that ROUTING OPEN SHOP with unit processing times is solvable in $2^{O(|V||M|^2 \log |V||M|)} \cdot \text{poly}(|J|)$ time, that is, *fixed-parameter tractable*.

Fixed-parameter algorithms. Fixed-parameter algorithms are an approach towards efficiently and optimally solving NP-hard problems: the main idea is to accept the exponential running time for finding optimal solutions to NP-hard problems, yet to confine it to some smaller problem parameter k (Cygan et al., 2015; Downey and Fellows, 2013; Flum and Grohe, 2006; Niedermeier, 2006). A problem with parameter k is called *fixed-parameter tractable (FPT)* if there is an algorithm that solves any instance I in $f(k) \cdot \text{poly}(|I|)$ time, where f is an arbitrary computable function. The corresponding algorithm is called *fixed-parameter algorithm*. In contrast to algorithms that merely run in polynomial time for fixed k , fixed-parameter algorithms can potentially solve NP-hard problems optimally and efficiently if the parameter k is small.

Recently, the field of fixed-parameter algorithmics has shown increased interest in scheduling (van Bevern et al., 2015a,c, 2016a,b; Bodlaender and Fellows, 1995; Fellows and McCartin, 2003; Halldórsson and Karlsson, 2006; Hermelin et al., 2015; Mnich and Wiese, 2015) and routing (van Bevern et al., 2014, 2015b; Dorn et al., 2013; Gutin et al., 2013, 2014a,b, 2015; Klein and Marx, 2014; Sorge et al., 2011, 2012), whereas fixed-parameter algorithms for problems containing elements of both routing and scheduling are still rare (Böckenhauer et al., 2007).

Our results. Using a variant of Galvin’s theorem on list-coloring edges of bipartite graphs (Borodin et al., 1997; Galvin, 1995), in Section 3 we prove a sufficient condition for the polynomial-time completability of partial schedules, which do not necessarily assign start times to all jobs on all machines, into feasible schedules.

We use the schedule completion theorem to prove upper bounds on various parameters of optimal schedules, in particular on their lengths in Section 4.

Using these bounds and integer linear programming, in Section 5 we show that ROUTING OPEN SHOP with unit processing times is fixed-parameter tractable parameterized by $|V| + |M|$. Note that, for arbitrary processing times, this is impossible unless $P = NP$.

Since the schedule extension theorem is a useful tool for proving upper bounds on various parameters of optimal schedules, we expect the schedule completion theorem to be likewise beneficial for approximation algorithms.

Input encoding. In general, a ROUTING OPEN SHOP instance requires at least $\Omega(|J| \cdot |M| + |E|)$ bits in order to encode the processing time of each job on each machine and the travel time for each edge. We call this the *standard encoding*. In contrast, an instance of ROUTING OPEN SHOP with unit processing times can be encoded using $O(|V|^2 \cdot \log c_{\max} + |V| \cdot \log |J|)$ bits by simply associating with each vertex in V the number of jobs it contains, where c_{\max} is the maximum travel time. We call this the *compact encoding*.

All running times in this article are stated for computing and outputting a minimum-length schedule, whose encoding requires at least $\Omega(|J| \cdot |M|)$ bits for the start time of each job on each machine. Thus, outputting the schedule is impossible in time polynomial in the size of the compact encoding. We therefore assume to get the input instance in standard encoding, like for general ROUTING OPEN SHOP.

However, we point out that the *decision version* of ROUTING OPEN SHOP with unit processing times is fixed-parameter tractable parameterized by $|V| + |M|$ even when assuming the compact encoding: our algorithm is able to decide whether there *exists* a schedule of given length L in $2^{O(|V||M|^2 \log |V||M|)} \cdot \text{poly}(|I|)$ time, where $|I|$ is the size an instance I given in compact encoding. To this end, the algorithm does not apply the schedule completion Theorem 3.4 to explicitly *construct* a schedule but merely to conclude its existence.

2. Preprocessing for metric travel times

In this section, we show how any instance can be transformed into an equivalent instance with travel times satisfying the triangle inequality. This will allow us to assume that, in an optimal schedule, a machine only stays in a vertex if it processes at least one job there: otherwise, it could take a “shortcut”, bypassing the vertex.

Lemma 2.1. Let I be a ROUTING OPEN SHOP instance and I' be obtained from I by replacing the graph $\mathcal{G} = (V, E)$ with travel times $c: E \rightarrow \mathbb{N}$ by a complete graph \mathcal{G}' on the vertex set V

with travel times $c' : \{v, w\} \mapsto \text{dist}_c(v, w)$, where $\text{dist}_c(v, w)$ is the length of a shortest path between v and w in \mathcal{G} with respect to c .

Then, any schedule for I is a schedule of the same length for I' and vice versa. Moreover, c' satisfies the triangle inequality $c'(\{v, w\}) \leq c'(\{v, u\}) + c'(\{u, w\})$ for all $u, v, w \in V$ and can be computed in $O(|V|^3)$ time.

Proof. It is obvious that c' satisfies the triangle inequality. It can be computed in $O(|V|^3)$ time using the Floyd-Warshall algorithm (Floyd, 1962).

Any feasible schedule for I is also a feasible schedule for I' of the same length since any route R for I is also a route for I' : for two consecutive stays (a_i, v_i, b_i) and $(a_{i+1}, v_{i+1}, b_{i+1})$ of R , one has $b_i + c'(v_i, v_{i+1}) \leq b_i + c(v_i, v_{i+1}) \leq a_{i+1}$.

Any feasible schedule for I' is a feasible schedule of the same length for I since any route R' with s stays for I' can be turned into a route of the same length with additional stays for I : for each $i \in \{1, \dots, s-1\}$, take two consecutive stays (a_i, v_i, b_i) and $(a_{i+1}, v_{i+1}, b_{i+1})$ on R' and a shortest path $P = (w_1 = v_i, w_2, \dots, w_\ell = v_{i+1})$ between v_i and v_{i+1} in \mathcal{G} with respect to c . Between stay i and $i+1$, add zero-length stays in the vertices of P . That is, for each $k \in \{1, \dots, \ell-2\}$, add stays

$$(a_i + \sum_{j=1}^k c(w_j, w_{j+1}), \quad w_{k+1}, \quad a_i + \sum_{j=1}^k c(w_j, w_{j+1}))$$

to R' . This yields a route R for I since

$$a_i + \sum_{j=1}^k c(w_j, w_{j+1}) + c(w_{k+1}, w_{k+2}) \leq a_i + \sum_{j=1}^{k+1} c(w_j, w_{j+1})$$

for all $k \in \{1, \dots, \ell-1\}$. Moreover, R has the same length as R' since the end of the last stay has not changed. \square

The main advantage of working on instances satisfying the triangle inequality is that we may assume that machines in an optimal schedule do not stay in vertices without processing jobs in them, except for the depot, which is always the first and last stay of a machine.

Lemma 2.2. Let S be a feasible schedule of length L for a ROUTING OPEN SHOP instance satisfying the triangle inequality.

Then, S is feasible with respect to machine routes $(R_{M_q})_{M_q \in \mathcal{M}}$ of length at most L such that, for each route $R = ((a_k, v_k, b_k))_{k=1}^s$ and each stay (a_k, v_k, b_k) on R , except, maybe, for $k \in \{1, s\}$, there is a job $J_i \in \mathcal{J}$ with $S(J_i, M_q) \in [a_k, b_k]$.

Proof. Since S is a feasible schedule of length L , it is feasible with respect to machine routes $(R_{M_q})_{M_q \in \mathcal{M}}$ of length L . Assume that each machine route R_{M_q} is minimal, that is, no stay can be removed without violating the feasibility of S with respect to R_{M_q} .

For the sake of contradiction, assume that the route $R_{M_q} = ((a_k, v_k, b_k))_{k=1}^s$ contains a stay (a_k, v_k, b_k) , where $k \notin \{1, s\}$, such that there is no job $J_i \in \mathcal{J}$ with $S(J_i, M_q) \in [a_k, b_k]$. Then,

removing (a_k, v_k, b_k) from R_{M_q} would yield a route R'_{M_q} with fewer stays since

$$\begin{aligned} b_{k-1} + c(v_{k-1}, v_{k+1}) &\leq b_{k-1} + c(v_{k-1}, v_k) + c(v_k, v_{k+1}) \\ &\leq a_k + c(v_k, v_{k+1}) \leq b_k + c(v_k, v_{k+1}) \leq a_{k+1}. \end{aligned}$$

Since S is feasible with respect to R'_{M_q} , this contradicts R_{M_q} being minimal. \square

Clearly, from Lemma 2.2, we get the following:

Observation 2.3. Vertices $v \in V \setminus \{v^*\}$ with $\mathcal{J}_v = \emptyset$ can be deleted from a ROUTING OPEN SHOP instance satisfying the triangle inequality, where v^* is the depot.

From now on, we assume that our input instances of ROUTING OPEN SHOP satisfy the triangle inequality and exploit Lemma 2.2 and Observation 2.3.

3. Schedule completion theorem

In this section, we present a theorem that allows us to complete *partial schedules*, which do not necessarily assign a start point to each job on each machine, into feasible schedules.

In the following, we consider only ROUTING OPEN SHOP with unit processing times and say that a machine M_q processes a job J_i at time $S(J_i, M_q)$ if it processes job J_i in the time interval $[S(J_i, M_q), S(J_i, M_q) + 1)$. We use $S(J_i, M_q) = \perp$ to denote that the processing time of job J_i on machine M_q is undefined.

Definition 3.1 (Partial schedule). A *partial schedule* with respect to given routes $(R_{M_q})_{M_q \in \mathcal{M}}$ of length at most L is a *partial function* $S : \mathcal{J} \times \mathcal{M} \rightarrow \mathbb{N}$ satisfying Definition 1.1(i–iii) for those jobs $J_i, J_j \in \mathcal{J}$ and machines $M_q, M_r \in \mathcal{M}$ for which $S(J_i, M_q) \neq \perp$ and $S(J_j, M_r) \neq \perp$. For a partial schedule $S : \mathcal{J} \times \mathcal{M} \rightarrow \mathbb{N}$, we introduce the following terminology:

$\mathcal{J}_{M_q}^S := \{J_i \in \mathcal{J} \mid S(J_i, M_q) = \perp\}$ is the set of jobs that lack processing by machine M_q ,

$\mathcal{M}_{J_i}^S := \{M_q \in \mathcal{M} \mid S(J_i, M_q) = \perp\}$ is the set of machines that job J_i lacks processing of (note that $J_i \in \mathcal{J}_{M_q}^S$ if and only if $M_q \in \mathcal{M}_{J_i}^S$),

$\mathcal{T}_{J_i}^S := \{t \leq L \mid \exists M_q \in \mathcal{M} : S(J_i, M_q) = t\}$ is the set of time units where job J_i is being processed,

$\mathcal{T}_{M_q}^S := \{t \leq L \mid \exists J_i \in \mathcal{J} : S(J_i, M_q) = t\}$ is the set of time units where machine M_q is processing,

$\mathcal{T}_v^{R_{M_q}} := \{t \leq L \mid R_{M_q} \text{ has a stay } (a_i, v, b_i) \text{ with } a_i \leq t < b_i\}$ are the time units where M_q stays in a vertex $v \in V$, and

$\mathcal{J}_v := \{J_i \in \mathcal{J} \mid \mathcal{L}(J_i) = v\}$ is the set of jobs in vertex $v \in V$ of \mathcal{G} .

The schedule completion theorem will allow us to turn any *completable* partial schedule into a feasible schedule. Intuitively, a schedule is completable if a machine has enough “free time” in each vertex to process all yet unprocessed jobs and to wait for other machines in the vertex to free the jobs to be processed.

Definition 3.2 (Completable schedule). Let $(R_{M_q})_{M_q \in \mathcal{M}}$ be a family of routes and, for each vertex $v \in V$, let $\bigcup_{s=1}^{g_v} \mathcal{M}_v^s := \mathcal{M}$ be a partition of machines such that, for any two machines $M_q \in \mathcal{M}_v^s$ and $M_r \in \mathcal{M}_v^t$ with $s \neq t$, one has $\mathcal{T}_v^{R_{M_q}} \cap \mathcal{T}_v^{R_{M_r}} = \emptyset$.

A partial schedule $S : \mathcal{J} \times \mathcal{M} \rightarrow \mathbb{N}$ with respect to $(R_{M_q})_{M_q \in \mathcal{M}}$ is *completable* if, for each vertex $v \in V$, each $1 \leq s \leq g_v$, each machine $M_q \in \mathcal{M}_v^s$, and each job $J_i \in \mathcal{J}_v^s \cap \mathcal{J}_v$, it holds that

$$|\mathcal{T}_v^{R_{M_q}} \setminus (\mathcal{T}_{J_i}^S \cup \mathcal{T}_{M_q}^S)| \geq \max\{|\mathcal{J}_{M_q}^S \cap \mathcal{J}_v|, |\mathcal{M}_{J_i}^S \cap \mathcal{M}_v^s|\}. \quad (3.1)$$

Example 3.3. Let $(R_{M_q})_{M_q \in \mathcal{M}}$ be routes such that all machines are in the same vertex at the same time, that is, $\mathcal{T}_v^{R_{M_q}} = \mathcal{T}_v^{R_{M_r}}$ for all vertices $v \in V$ and machines $M_q, M_r \in \mathcal{M}$. Moreover, assume that each machine $M_q \in \mathcal{M}$ stays in each vertex $v \in V$ at least $\max\{|\mathcal{J}_v|, |\mathcal{M}|\}$ time, that is, $|\mathcal{T}_v^{R_{M_q}}| \geq \max\{|\mathcal{J}_v|, |\mathcal{M}|\}$. Then the empty schedule is completable and, by the following schedule completion theorem, there is a feasible schedule with respect to the routes $(R_{M_q})_{M_q \in \mathcal{M}}$.

Theorem 3.4 (Schedule completion theorem). Given a partial schedule $S : \mathcal{J} \times \mathcal{M} \rightarrow \mathbb{N}$ that is completable with respect to routes $(R_{M_q})_{M_q \in \mathcal{M}}$, one can compute a feasible schedule $S' \supseteq S$ with respect to the routes $(R_{M_q})_{M_q \in \mathcal{M}}$ in time polynomial in $|\mathcal{J}| + |\mathcal{M}| + |V| + \sum_{v \in V, M_q \in \mathcal{M}} |\mathcal{T}_v^{R_{M_q}}|$.

We prove Theorem 3.4 using a stronger version of Galvin's theorem about properly list-coloring the edges of bipartite graphs (Borodin et al., 1997; Galvin, 1995).

Definition 3.5 (Proper edge coloring, f -edge-choosable). By $\deg(v)$, we denote the *degree* of a vertex v , that is, the number of edges incident to v .

A *proper edge coloring* of a graph $G = (V, E)$ is a coloring $C : E \rightarrow \mathbb{N}$ of the edges of G such that $C(e_1) \neq C(e_2)$ if $e_1 \cap e_2 \neq \emptyset$, that is, if e_1 and e_2 share a vertex.

A graph $G = (V, E)$ is *f -edge-choosable* for some function $f : E \rightarrow \mathbb{N}$ if G allows for a proper edge coloring $C : E \rightarrow \mathbb{N}$ with $C(e) \in L_e$ for every family $\{L_e \subseteq \mathbb{N} \mid e \in E\}$ with $|L_e| \geq f(e)$.

Theorem 3.6 (Borodin et al. (1997)). Any bipartite graph $G = (V, E)$ is f -edge-choosable for

$$f : E \rightarrow \mathbb{N}, \{u, v\} \mapsto \max\{\deg(u), \deg(v)\}.$$

Remark 3.7. The proof given by Borodin et al. (1997) is constructive: given a bipartite graph $G = (V, E)$ and a set $L_e \subseteq \mathbb{N}$ with $|L_e| \geq f(e)$ for each edge $e \in E$, a proper edge coloring $C : E \rightarrow \mathbb{N}$ with $C(e) \in L_e$ is computable in time polynomial in the size of G and the color sets.

Before Borodin et al. (1997) proved Theorem 3.6, Galvin (1995) proved the special case for $f : e \mapsto \Delta$, where Δ is the maximum degree of G . Before Galvin's proof, its special case with $G = K_{n,n}$ being a complete bipartite graph and $f : e \mapsto n$ was known as Dinitz' conjecture. We now use Theorem 3.6 to prove Theorem 3.4.

Proof of Theorem 3.4. Let $B = (\mathcal{J} \cup \mathcal{M}, X)$ be a bipartite graph with an edge $\{J_i, M_q\} \in X$ if and only if $S(J_i, M_q) = \perp$ for $J_i \in \mathcal{J}$ and $M_q \in \mathcal{M}$. We compute a proper edge coloring C of B such that, for each edge $\{J_i, M_q\} \in X$, we have

$$C(\{J_i, M_q\}) \in \mathcal{T}_{\mathcal{J}_i}^{R_{M_q}} \setminus (\mathcal{T}_{J_i}^S \cup \mathcal{T}_{M_q}^S) \quad (3.2)$$

$$\text{and define } S'(J_i, M_q) := \begin{cases} C(\{J_i, M_q\}) & \text{if } \{J_i, M_q\} \in X \text{ and} \\ S(J_i, M_q) & \text{otherwise.} \end{cases}$$

It remains to show that

- (1) the edge coloring C is computable in time polynomial in $|\mathcal{J}| + |\mathcal{M}| + |V| + \sum_{v \in V, M_q \in \mathcal{M}} |\mathcal{T}_v^{R_{M_q}}|$ and that
- (2) S' is a feasible schedule.

(1) We obtain the proper edge coloring C by independently computing a proper edge coloring C_{v_s} satisfying (3.2) for each induced subgraph $B_{v_s} := B[\mathcal{J}_v \cup \mathcal{M}_v^s]$ for all $v \in V$ and $1 \leq s \leq g_v$. To this end, observe that, in B_{v_s} , a vertex $J_i \in \mathcal{J}_v$ has degree $|\mathcal{M}_{J_i}^s \cap \mathcal{M}_v^s|$ and that a vertex $M_q \in \mathcal{M}_v^s$ has degree $|\mathcal{J}_{M_q}^s \cap \mathcal{J}_v|$. Thus, by Theorem 3.6, if, for each edge $e := \{J_i, M_q\}$ of B_{v_s} , we have a list L_e of colors with $|L_e| \geq \max\{|\mathcal{J}_{M_q}^s \cap \mathcal{J}_v|, |\mathcal{M}_{J_i}^s \cap \mathcal{M}_v^s|\}$, then B_{v_s} has a proper edge coloring C_{v_s} with $C_{v_s}(e) \in L_e$ for each edge e of B_{v_s} . Since S is completable (Definition 3.2), simply choosing $L_{\{J_i, M_q\}} := \mathcal{T}_v^{R_{M_q}} \setminus (\mathcal{T}_{J_i}^S \cup \mathcal{T}_{M_q}^S)$ for each edge $\{J_i, M_q\}$ of B_{v_s} yields a proper edge coloring C_{v_s} for B_{v_s} satisfying (3.2).

We now let $C := \bigcup_{v \in V, 1 \leq s \leq g_v} C_{v_s}$. This is a proper edge coloring for the bipartite graph B since, for edges e_{v_s} of B_{v_s} and e_{w_t} of B_{w_t} with $v \neq w$ or $s \neq t$, we have $L_{e_{v_s}} \cap L_{e_{w_t}} = \emptyset$: for any vertex $v \in V$ and machines $M_q \in \mathcal{M}_v^s, M_r \in \mathcal{M}_v^t$ with $s \neq t$, one has $\mathcal{T}_v^{R_{M_q}} \cap \mathcal{T}_v^{R_{M_r}} = \emptyset$, and for any machine $M_q \in \mathcal{M}$ and $v \neq w \in V$, one has $\mathcal{T}_v^{R_{M_q}} \cap \mathcal{T}_w^{R_{M_q}} = \emptyset$. Moreover, C satisfies (3.2) since each C_{v_s} for $v \in V$ and $1 \leq s \leq g_v$ satisfies (3.2).

Regarding the running time, it is clear that, for each $v \in V$ and $1 \leq s \leq g_v$, the bipartite graph B_{v_s} and the sets $L_{\{J_i, M_q\}}$ of allowed colors for each edge $\{J_i, M_q\}$ are computable in time polynomial in $|\mathcal{J}| + |\mathcal{M}| + |\mathcal{T}_v^{R_{M_q}}|$.³ Moreover, by Remark 3.7, the sought edge coloring C_{v_s} for each B_{v_s} is computable in time polynomial in $|B_{v_s}| + \sum_{e \in E(B_{v_s})} |L_e|$.

(2) We first show that S' is a schedule. For each job $J_i \in \mathcal{J}$ and each machine $M_q \in \mathcal{M}$ we have $S(J_i, M_q) \neq \perp$ or an edge $\{J_i, M_q\} \in X$ in B . Thus, $S'(J_i, M_q) = S(J_i, M_q) \neq \perp$ or $S'(J_i, M_q) = C(\{J_i, M_q\}) \neq \perp$ and S' is a schedule. We show that S' is feasible.

First, let $J_i \in \mathcal{J}$ be a job and $M_q, M_r \in \mathcal{M}$ be distinct machines. We show that $S'(J_i, M_q) \neq S'(J_i, M_r)$. We distinguish three cases. If $S(J_i, M_q) \neq \perp$ and $S(J_i, M_r) \neq \perp$, then $S'(J_i, M_q) = S(J_i, M_q) \neq S(J_i, M_r) = S'(J_i, M_r)$. If $S(J_i, M_q) = \perp = S(J_i, M_r)$, then $S'(J_i, M_q) = C(\{J_i, M_q\}) \neq C(\{J_i, M_r\}) = S'(J_i, M_r)$. Finally, if $S(J_i, M_q) = \perp$ and $S(J_i, M_r) \neq \perp$, then $S'(J_i, M_q) \neq S'(J_i, M_r)$ since $S'(J_i, M_q) = C(\{J_i, M_q\}) \in \mathcal{T}_v^{R_{M_q}} \setminus (\mathcal{T}_{J_i}^S \cup \mathcal{T}_{M_q}^S)$ and $S(J_i, M_r) \in \mathcal{T}_{J_i}^S$.

³We abstain from a more detailed running time analysis since no such analysis is available for the forthcoming application of Theorem 3.6 (yet).

Now, let $J_i, J_j \in \mathcal{J}$ be two distinct jobs and $M_q \in \mathcal{M}$ be a machine. We show that $S'(J_i, M_q) \neq S'(J_j, M_q)$. We distinguish three cases. If $S(J_i, M_q) \neq \perp$ and $S(J_j, M_q) \neq \perp$, then $S'(J_i, M_q) = S(J_i, M_q) \neq S(J_j, M_q) = S'(J_j, M_q)$. If $S(J_i, M_q) = \perp = S(J_j, M_q)$, then $S'(J_i, M_q) = C(\{J_i, M_q\}) \neq C(\{J_j, M_q\}) = S'(J_j, M_q)$. Finally, if we have $S(J_i, M_q) = \perp$ and $S(J_j, M_q) \neq \perp$, then $S'(J_i, M_q) \neq S'(J_j, M_q)$ since $S'(J_i, M_q) = C(\{J_i, M_q\}) \in \mathcal{T}_v^{R_{M_q}} \setminus (\mathcal{T}_{J_i}^S \cup \mathcal{T}_{M_q}^S)$ and $S'(J_j, M_q) \in \mathcal{T}_{M_q}^S$. \square

4. Upper and lower bounds

In this section, we show lower and upper bounds on the lengths of optimal solutions to ROUTING OPEN SHOP with unit processing times. These will be exploited in our fixed-parameter algorithm and make first steps towards approximation algorithms.

We assume ROUTING OPEN SHOP instances to be preprocessed to satisfy the triangle inequality. By Lemma 2.1, this does not change the length of optimal schedules. However, it ensures that the minimum cost of a cycle visiting each vertex of the graph $\mathcal{G} = (V, E)$ with travel times $c: E \rightarrow \mathbb{N}$ at least once coincides with the minimum cost of a cycle doing so *exactly once* (Serdyukov, 1978), that is, of a *Hamiltonian cycle*.

A simple lower bound is given by the fact that, in view of Observation 2.3, all machines have to visit each vertex at least once and have to process $|\mathcal{J}|$ jobs.

Observation 4.1. Let H be a minimum-cost Hamiltonian cycle in the graph $\mathcal{G} = (V, E)$ with metric travel times $c: E \rightarrow \mathbb{N}$. Then, any feasible schedule has length at least $c(H) + |\mathcal{J}|$.

A trivial upper bound can be given by letting the machines work sequentially.

Observation 4.2. Given a Hamiltonian cycle H in the graph $\mathcal{G} = (V, E)$ with travel times $c: E \rightarrow \mathbb{N}$, a feasible schedule of length $c(H) + |\mathcal{J}| + |\mathcal{M}| - 1$ is computable in $O(|\mathcal{J}| \cdot |\mathcal{M}| + |V|)$ time.

This bound can be improved if $c(H) \leq \min\{|\mathcal{J}|, |\mathcal{M}|\} - 1$

Proposition 4.3. Given a Hamiltonian cycle H in the graph $\mathcal{G} = (V, E)$ with travel times $c: E \rightarrow \mathbb{N}$, a feasible schedule of length $2c(H) + \max\{|\mathcal{J}|, |\mathcal{M}|\}$ is computable in $O(|\mathcal{J}|^2 + |\mathcal{M}| + |V|)$ time.

Proof. Let $n := |\mathcal{J}|$ and $m := |\mathcal{M}|$. Without loss of generality, assume that $n \geq m$. Otherwise, we can simply add $m - n$ additional jobs to the depot and finally remove them from the constructed schedule. We will construct a feasible schedule S of length $2c(H) + n$ by constructing a matrix $S = (s_{iq})_{1 \leq i \leq n, 1 \leq q \leq m}$, where s_{iq} determines the time at which job J_i is processed by machine M_q .

Let $H = (v_1, v_2, \dots, v_{|V|})$, where $v_1 = v^*$ is the depot. Without loss of generality, let the jobs J_1, \dots, J_n be ordered so that, for jobs J_i, J_j with $i \leq j$, one has $J_i \in \mathcal{J}_{v_k}$ and $J_j \in \mathcal{J}_{v_\ell}$ with $k \leq \ell$. That is, the first jobs are in v_1 , then follow jobs in v_2 , and so on. First, construct a matrix $S' = (s'_{iq})_{1 \leq i \leq n, 1 \leq q \leq m}$ with

$$s'_{iq} := (i - q) \bmod n = \begin{cases} n - q + i & \text{if } i < q, \\ i - q & \text{otherwise.} \end{cases}$$

Call a cell s'_{iq} *red* if $i < q$ and *green* otherwise. Note that if s'_{iq} and s'_{jr} are of the same color and $i < j$ or $r < q$, then $s'_{iq} < s'_{jr}$. Moreover, the number in a red cell is larger than the number in any green cell of the same row or column: if s'_{iq} is red and s'_{jq} is green, then from

$$n + i > j \quad \text{follows} \\ s'_{iq} = n - q + i > j - q = s'_{jq}$$

and if s'_{iq} is red and s'_{ir} is green, then from

$$n - q > -r \quad \text{follows} \\ s'_{iq} = n - q + i > i - r = s'_{ir}.$$

Let $c_k = \sum_{i=2}^k c(v_{i-1}, v_i)$ be the travel time from v_1 to v_k along H . Clearly, the sequence $(c_k)_{1 \leq k \leq s}$ is non-decreasing and $c_{|V|} \leq c(H)$. Our schedule is now given by $S = (s_{iq})_{1 \leq i \leq n, 1 \leq q \leq m}$, where

$$s_{iq} := s'_{iq} + \begin{cases} c_k & \text{if } \mathcal{L}(J_i) = v_k \text{ and } s_{iq} \text{ is green,} \\ c(H) + c_k & \text{if } \mathcal{L}(J_i) = v_k \text{ and } s_{iq} \text{ is red,} \end{cases}$$

and $\mathcal{L}(J_i)$ is the vertex where job J_i is located.

Let us prove that this schedule is feasible in terms of Definition 1.1. Indeed, by construction, for two elements s_{iq} and s_{jr} with $i = j$ or $q = r$ and $s'_{iq} > s'_{jr}$, one has $s_{iq} > s_{jr}$ since the value added to s'_{iq} is not smaller than the value added to s'_{jr} due to our sorting of jobs by non-decreasing vertex indices and because the value added to any red cell is larger than any value added to a green cell. Therefore, conditions (i) and (ii) are satisfied.

It remains to determine the routes $R_{M_q} = ((a_i, v_i, b_i))_{i=1}^t$ for each machine $M_q \in \mathcal{M}$. Machine M_q will follow H up to two times. During the first stay (a_k^1, v_k, b_k^1) in a vertex v_k , it will process all jobs J_i such that s_{iq} is green. During the second stay (a_k^2, v_k, b_k^2) , it will process all jobs J_i such that s_{iq} is red. That is, the beginning and end times of the stays are

$$\begin{aligned} a_k^1 &:= \min\{S(J_i, M_q) \mid J_i \in \mathcal{J}, \mathcal{L}(J_i) = v_k \text{ and } s_{ij} \text{ is green}\}, \\ b_k^1 &:= \max\{S(J_i, M_q) \mid J_i \in \mathcal{J}, \mathcal{L}(J_i) = v_k \text{ and } s_{ij} \text{ is green}\} + 1, \\ a_k^2 &:= \min\{S(J_i, M_q) \mid J_i \in \mathcal{J}, \mathcal{L}(J_i) = v_k \text{ and } s_{ij} \text{ is red}\}, \\ b_k^2 &:= \max\{S(J_i, M_q) \mid J_i \in \mathcal{J}, \mathcal{L}(J_i) = v_k \text{ and } s_{ij} \text{ is red}\} + 1. \end{aligned}$$

By the choice of s_{iq} for red cells, the machines have enough time to go around H a second time. It is thus easy to verify that the chosen routes satisfy the condition (iii) and that the length of the schedule is at most $n + 2c(H)$. \square

We next study for which instances one gets an upper bound that matches the lower bound from Observation 4.1. In Example 3.3, we have already seen that arbitrary machine routes that stay in each vertex v at least $\max\{|\mathcal{J}_v|, |\mathcal{M}|\}$ time can be completed into a feasible schedule. We therefore distinguish vertices v for which staying $|\mathcal{J}_v|$ time is both necessary and sufficient.

Definition 4.4 (Criticality of vertices). For a vertex $v \in V$, we denote by

$$k(v) := \max\{0, |\mathcal{M}| - |\mathcal{J}_v|\} \text{ the criticality of } v, \text{ and by}$$

$$K := \sum_{v \in V} k(v) \text{ the total criticality.}$$

A vertex $v \in V$ is *critical* if $k(v) > 0$, that is, if $|\mathcal{J}_v| < |\mathcal{M}|$.

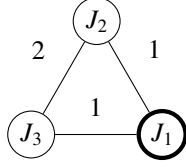


Figure 1: On the left: a graph with one job in each vertex, travel times as denoted on the edges, and the depot being J_1 . On the right: a schedule S of length 9 to process these jobs on seven machines. Note that machine M_7 does not travel along a Hamiltonian cycle, but along route J_1, J_2, J_1, J_3, J_1 . One can show that any schedule in which machines travel along Hamiltonian cycles has length at least 10.

Proposition 4.5. Given a Hamiltonian cycle H in the graph $\mathcal{G} = (V, E)$ with travel times $c: E \rightarrow \mathbb{N}$, a feasible schedule of length at most $c(H) + |\mathcal{J}| + K$ can be computed in polynomial time.

Proof. Let $H = (v_1, v_2, \dots, v_{|V|})$ and $v_{|V|+1} := v_1$. Without loss of generality, assume that $v_1 = v^*$ is the depot. Each machine $M_q \in \mathcal{M}$ uses the same route R of $|V| + 1$ stays $(a_1, v_1, b_1), \dots, (a_{|V|+1}, v_{|V|+1}, b_{|V|+1})$, where

$$\begin{aligned} a_1 &:= 0, \\ b_i &:= a_i + |\mathcal{J}_{v_i}| + k(v_i), & \text{for } i \in \{1, \dots, |V|\}, \\ a_{i+1} &:= b_i + c(v_i, v_{i+1}), & \text{for } i \in \{1, \dots, |V|\}, \\ b_{|V|+1} &:= a_{|V|+1}. \end{aligned}$$

Each stay (a_i, v_i, a_i) lasts $|\mathcal{J}_{v_i}| + k(v_i) = \max\{|\mathcal{J}_{v_i}|, |\mathcal{M}|\}$ time. By Theorem 3.4, the empty schedule S is completable into a feasible schedule S' with respect to the route R for each machine and S' is computable in time polynomial in $|\mathcal{J}| + |\mathcal{M}| + \sum_{v \in V} |\mathcal{T}_v^R| \in O(|\mathcal{J}| + |\mathcal{M}| + K) \subseteq O(|\mathcal{J}| + |\mathcal{M}| + |V| \cdot |\mathcal{M}|)$. Finally, the route R has length

$$\begin{aligned} b_{|V|+1} = a_{|V|+1} &= \sum_{i=1}^{|V|} (|\mathcal{J}_{v_i}| + k(v_i) + c(v_i, v_{i+1})) \\ &= c(H) + |\mathcal{J}| + K. \end{aligned} \quad \square$$

Combining Observation 4.1 and Proposition 4.5 and that a minimum-cost Hamiltonian cycle can be computed in $O(2^{|V|} \cdot |V|^2)$ time using the algorithm of Bellman (1962), Held and Karp (1962), we obtain a first fixed-parameter tractability result:

Corollary 4.6. ROUTING OPEN SHOP with unit processing times is fixed-parameter tractable parameterized by $|V|$ if there are no critical vertices.

Corollary 4.6 makes clear that, given the schedule completion theorem, critical vertices are the main obstacle for solving ROUTING OPEN SHOP with unit processing times: while staying $|\mathcal{J}_v|$ time in a non-critical vertex $v \in V$ is both necessary and sufficient, staying in critical vertices $|\mathcal{M}|$ time is sufficient, but not necessary. Indeed, as shown in Figure 1, in the presence of critical vertices, there might not even be optimal schedules in which the machines travel along Hamiltonian cycles.

5. Fixed-parameter algorithm

In this section, we present a fixed-parameter algorithm for ROUTING OPEN SHOP with unit processing times, which is our main algorithmic result:

Theorem 5.1. ROUTING OPEN SHOP with unit processing times is solvable in $2^{O(|V||\mathcal{M}|^2 \log |V||\mathcal{M}|)} \cdot \text{poly}(|\mathcal{J}|)$ time.

The outline of the algorithm for Theorem 5.1 is as follows: in Section 5.1, we use the schedule completion Theorem 3.4 to show that the routes of a minimum-length schedule comply with one of $2^{O(|V||\mathcal{M}|^2 \log |V||\mathcal{M}|)}$ *pre-schedules*, which determines the sequence of vertices that each machine stays in, the durations of stays in critical vertices, and the time offsets between stays in critical vertices.

In Section 5.2, we use integer linear programming to compute, for each pre-schedule, shortest complying routes so that each machine stays in each non-critical vertex v for at least $|\mathcal{J}_v|$ time. The schedule for non-critical vertices is then implied by the schedule completion Theorem 3.4, whereas we compute the schedule for critical vertices using brute force.

5.1. Enumerating pre-schedules

We will show that the routes of a minimum-length schedule comply with *some pre-schedule*, which is defined below and illustrated in Figure 2.

Definition 5.2 (Pre-schedule). A *pre-stay* is a triple $(M_q, v, \sigma) \in \mathcal{M} \times V \times \{1, \dots, |V||\mathcal{M}| + 2\}$, intuitively meaning that a machine $M_q \in \mathcal{M}$ has its σ -th stay in vertex $v \in V$. We call $T = ((M_{q_i}, v_i, \sigma_i))_{i=1}^s$ a *pre-stay sequence* if,

- (i) for each $M_q \in \mathcal{M}$, the σ_i with $q_i = q$ increase in steps of one for increasing i .

Machine routes $(R_{M_q})_{M_q \in \mathcal{M}}$, where $R_{M_q} = ((a_k^q, w_k^q, b_k^q))_{k=1}^{t_q}$, comply with a pre-stay sequence if

- (ii) route R_{M_q} has a stay (a_k^q, w_k^q, b_k^q) if and only if (M_q, w_k^q, k) is in T and,
- (iii) for pre-stays (M_{q_i}, v_i, σ_i) and (M_{q_j}, v_j, σ_j) with $i < j$, one has $a_{\sigma_i}^{q_i} \leq a_{\sigma_j}^{q_j}$.

Let $\mathcal{K} := \{i \leq s \mid v_i \text{ is critical}\}$ be the indices of pre-stays in critical vertices of T . A *length assignment* is a map $A: \mathcal{K} \rightarrow \{0, \dots, 2|\mathcal{M}| - 1\}$. Machine routes $(R_{M_q})_{M_q \in \mathcal{M}}$ comply with a length assignment A if,

- (iv) for each pre-stay (M_{q_i}, v_i, σ_i) in T with $i \in \mathcal{K}$, one has $b_{\sigma_i}^{q_i} - a_{\sigma_i}^{q_i} = A(i)$.

A *displacement* is a map $D: \mathcal{K} \rightarrow \{0, \dots, 2|\mathcal{M}|\}$. The machine routes $(R_{M_q})_{M_q \in \mathcal{M}}$ comply with a displacement D if

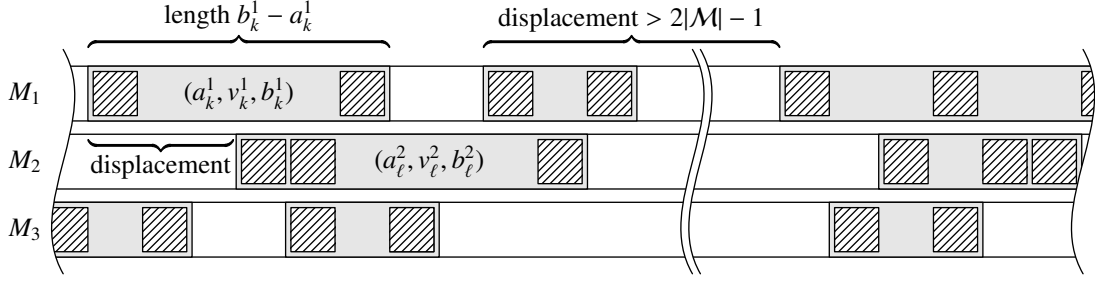


Figure 2: Shown is a part of a schedule for three machines M_1, M_2 , and M_3 . The horizontal axis is time. Uniformly gray boxes are stays in critical vertices that comply with some pre-schedule (the pre-schedule is not shown). Hatched squares correspond to jobs being processed. Illustrated are the lengths of stays and displacements between stays in critical vertices that are consecutive in the pre-stay sequence (stays in non-critical vertices are not shown). Herein, this displacement is either smaller than $2|M|$, in which case the relative position in time of stays is fixed for *any* complying set of machine routes, or at least $2|M|$, in which case the stays cannot intersect in time for *any* set of complying machine routes since stays in critical vertices have length at most $2|M| - 1$ by Definition 5.2(iv). This will allow us, without knowing the absolute start and end time of stays, to check the feasibility of partial schedules for stays in critical vertices in any set of complying machine routes.

- (v) for two pre-stays (M_{q_i}, v_i, σ_i) and (M_{q_j}, v_j, σ_j) such that $i, j \in \mathcal{K}$ and $k \notin \mathcal{K}$ for all $k \in \{i + 1, \dots, j - 1\}$, one has

$$\begin{aligned} a_{\sigma_j}^{q_j} &\geq a_{\sigma_i}^{q_i} + 2|M| && \text{if } D(j) = 2|M| \text{ and} \\ a_{\sigma_j}^{q_j} &= a_{\sigma_i}^{q_i} + D(j) && \text{if } D(j) < 2|M|. \end{aligned}$$

We call (T, A, D) a *pre-schedule* and say that machine routes *comply* with (T, A, D) if they comply with each of T, A , and D , that is, (i)–(v) hold.

We show that an optimal solution for ROUTING OPEN SHOP with unit processing times can be found by solving instances of the following problem:

Problem 5.3.

Input: An instance I of ROUTING OPEN SHOP with unit processing times, a pre-schedule (T, A, D) , and a natural number L .

Task: Compute a schedule whose machine routes $(R_{M_q})_{M_q \in \mathcal{M}}$ have length at most L and comply with (T, A, D) , if such a schedule exists.

Proposition 5.4. For a ROUTING OPEN SHOP instance I with unit processing times there is a set \mathcal{I} of $2^{O(|V||\mathcal{M}|^2 \log |V||\mathcal{M}|)}$ instances of Problem 5.3 such that

- (i) if some instance $(I, (T, A, D), L) \in \mathcal{I}$ has a solution S , then S is a schedule of length at most L for I and
- (ii) there is a minimum-length schedule S for I such that S is a solution for at least one instance $(I, (T, A, D), L) \in \mathcal{I}$, where L is the length of S .

The set \mathcal{I} can be generated in $2^{O(|V||\mathcal{M}|^2 \log |V||\mathcal{M}|)} \cdot \text{poly}(|\mathcal{J}|)$ time.

The proof of Proposition 5.4 is based on proving that there are at most $2^{O(|V||\mathcal{M}|^2 \log |V||\mathcal{M}|)}$ pre-schedules and that the routes of an optimal schedule comply with at least one pre-schedule.

We will use the following lemma to show that there is a pre-stay sequence that the routes of an optimal schedule comply with:

Lemma 5.5. Each of the routes $(R_{M_q})_{M_q \in \mathcal{M}}$ of an optimal schedule consists of at most $|V||\mathcal{M}| + 2$ stays.

Proof. Let H be a minimum-cost Hamiltonian cycle for the graph \mathcal{G} with travel times $c: E \rightarrow \mathbb{N}$. Let $M_q \in \mathcal{M}$ be an arbitrary machine. It has to stay in all vertices and return to the depot, that is, its tour R_{M_q} has at least $|V| + 1$ stays. Moreover, by Observation 4.1, its length is at least $c(H) + |\mathcal{J}|$. Since $c(e) \geq 1$ for each $e \in E$ (see Definition 1.1), each additional stay increases the length of the tour by at least one.

Thus, if R_{M_q} had more than $|V| + K + 1$ stays, where K is the total critically of vertices in the input instance (see Definition 4.4), then it would have length at least $c(H) + |\mathcal{J}| + K + 1$, contradicting the optimality of the schedule by Proposition 4.5. Thus, the number of stays on R_{M_q} is at most

$$\begin{aligned} |V| + K + 1 &= |V| + \sum_{v \in V} \max\{0, |\mathcal{M}| - |\mathcal{J}_v|\} + 1 \\ &\leq |V| + (|V| - 1)(|\mathcal{M}| - 1) + |\mathcal{M}| + 1 = |V||\mathcal{M}| + 2 \end{aligned}$$

since, by Observation 2.3, only for the depot v^* one might have $\mathcal{J}_{v^*} = \emptyset$. \square

We will use the following lemma to show that there are also length assignments and displacements that the routes of an optimal schedule comply with. For the notation used in Lemma 5.6, recall Definition 3.1.

Lemma 5.6. For each feasible schedule S with respect to machine routes $(R_{M_q})_{M_q \in \mathcal{M}}$, there is a feasible schedule S' of the same length with respect to machine routes $(R'_{M_q})_{M_q \in \mathcal{M}}$ such that $|\mathcal{T}_v^{R'_{M_q}}| \leq \max\{|\mathcal{J}_v|, |\mathcal{M}|\} + |\mathcal{M}| - 1$ for each vertex $v \in V$.

Proof. For each machine $M_q \in \mathcal{M}$, construct the route R'_{M_q} from the route $R_{M_q} = ((a_k, v_k, b_k))_{k=1}^s$ as follows:

1. If $|\mathcal{T}_v^{R_{M_q}}| \leq \max\{|\mathcal{J}_v|, |\mathcal{M}|\} + |\mathcal{M}| - 1$, then $R'_{M_q} := R_{M_q}$,
2. Otherwise, let $R'_{M_q} := ((a'_i, v_i, b'_i))_{i=1}^s$, where $a_i \leq a'_i \leq b'_i \leq b_i$ for $1 \leq i \leq s$ are chosen arbitrarily with $|\mathcal{T}_v^{R'_{M_q}}| = \max\{|\mathcal{J}_v|, |\mathcal{M}|\} + |\mathcal{M}| - 1$.

Denote by $\overline{\mathcal{M}} := \{M_q \in \mathcal{M} \mid R_{M_q} \neq R'_{M_q}\}$ the set of machines whose tours have been altered. If $\overline{\mathcal{M}} = \emptyset$, then there is nothing

to prove. Henceforth, assume $\overline{\mathcal{M}} \neq \emptyset$. Then, S might not be a feasible schedule for the routes $(R'_{M_q})_{M_q \in \mathcal{M}}$ but

$$S^*(J_i, M_q) := \begin{cases} \perp & \text{if } M_q \in \overline{\mathcal{M}}, \\ S(J_i, M_q) & \text{otherwise} \end{cases}$$

is a partial schedule for the routes $(R'_{M_q})_{M_q \in \mathcal{M}}$ since the machines in $\overline{\mathcal{M}}$ do not process any jobs in S^* . We show that S^* is completable with respect to $(R'_{M_q})_{M_q \in \mathcal{M}}$ in terms of Definition 3.2.

To this end, choose an arbitrary vertex $v \in V$ and an arbitrary machine $M_q \in \mathcal{M}$ with some unprocessed job $J_i \in \mathcal{J}_{M_q}^*$. Then, $M_q \in \overline{\mathcal{M}}$ since only machines in $\overline{\mathcal{M}}$ have unprocessed jobs in S^* . Moreover, $|\mathcal{J}_{J_i}^{S^*}| \leq |\mathcal{M}| - 1$, since at least the machine M_q does not process J_i . Finally $\mathcal{T}_{M_q}^{S^*} = \emptyset$ since M_q does not process any jobs in S^* . Thus,

$$\begin{aligned} |\mathcal{T}_v^{R'_{M_q}} \setminus (\mathcal{T}_{J_i}^{S^*} \cup \mathcal{T}_{M_q}^{S^*})| &\geq \max\{|\mathcal{J}_v|, |\mathcal{M}|\} + |\mathcal{M}| - 1 - (|\mathcal{M}| - 1) \\ &= \max\{|\mathcal{J}_v|, |\mathcal{M}|\} \end{aligned}$$

and Theorem 3.4 shows how to complete S^* into a feasible schedule S' for the routes $(R'_{M_q})_{M_q \in \mathcal{M}}$. \square

Remark 5.7. Lemma 5.6 gives an upper bound of $\max\{|\mathcal{J}_v|, |\mathcal{M}|\} + |\mathcal{M}| - 1$ on the total amount of time that each machine stays in a vertex v in an optimal schedule. Note that neither Example 3.3 nor Proposition 4.5 give such an upper bound: these show that, in order to obtain a feasible schedule, it is sufficient that each machine stays in each vertex v for *at least* $\max\{|\mathcal{J}_v|, |\mathcal{M}|\}$ time. They do not exclude that, in an optimal schedule, a machine might stay in a vertex significantly longer in order to enable other machines to process their jobs faster.

We are now ready to prove Proposition 5.4.

Proof of Proposition 5.4. (i) is trivially true by definition of Problem 5.3.

(ii) Let S be a schedule for I with respect to minimum-length machine routes $(R_{M_q})_{M_q \in \mathcal{M}}$. We choose $\mathcal{I} := \{(I, (T, A, D), L) \mid (T, A, D) \text{ is a pre-schedule and } c(H) + |\mathcal{J}| \leq L \leq c(H) + |\mathcal{J}| + K\}$, where H is a minimum-cost Hamiltonian cycle for \mathcal{G} and K is the total criticality (see Definition 4.4). We first show that the routes $(R_{M_q})_{M_q \in \mathcal{M}}$ comply with some pre-schedule (T, A, D) . Since, by Observation 4.1 and Proposition 4.5, the machine routes $(R_{M_q})_{M_q \in \mathcal{M}}$ have length $L \in [c(H) + |\mathcal{J}|, c(H) + |\mathcal{J}| + K]$, it follows that S is a solution for some $(I, (T, A, D), L) \in \mathcal{I}$. Thereafter, we analyze the cardinality of \mathcal{I} .

By Lemma 5.5, each route $R_{M_q} = ((a_k^q, w_k^q, b_k^q))_{k=1}^{t_q}$ of a machine $M_q \in \mathcal{M}$ has $t_q \leq |V||\mathcal{M}| + 2$ stays. Thus, by Definition 5.2(i-iii), they comply with the pre-stay sequence $T := ((M_{q_i}, v_i, \sigma_i))_{i=1}^s$ that has a pre-stay $(M_{q_i}, v_i, \sigma_i) = (M_q, w_k^q, k)$ if and only if R_{M_q} has a stay (a_k^q, w_k^q, b_k^q) , where T is sorted so that one has $a_{\sigma_i}^{q_i} \leq a_{\sigma_j}^{q_j}$ for $i < j$ and so that one has $\sigma_i < \sigma_j$ for $i < j$ with $q_i = q_j$. Such a sorting exists since $a_i^q \leq a_j^q$ for $i < j$ and all machines $M_q \in \mathcal{M}$. Now, let $\mathcal{K}_T := \{i \leq s \mid T \text{ has a pre-stay in a critical vertex } v_i\}$. By Lemma 5.6, we may assume that the routes $(R_{M_q})_{M_q \in \mathcal{M}}$ stay in a critical vertex at most $2|\mathcal{M}| - 1$ units

of time. Thus, by Definition 5.2(iv), the routes $(R_{M_q})_{M_q \in \mathcal{M}}$ comply with the length assignment $A: \mathcal{K}_T \rightarrow \{0, \dots, 2|\mathcal{M}| - 1\}$, $i \mapsto b_{\sigma_i}^{q_i} - a_{\sigma_i}^{q_i}$. It is trivial that the routes $(R_{M_q})_{M_q \in \mathcal{M}}$ comply with the displacement $D: \mathcal{K}_T \rightarrow \{0, \dots, 2|\mathcal{M}|\}$, $j \mapsto \min\{2|\mathcal{M}|, a_{\sigma_j}^{q_j} - a_{\sigma_i}^{q_i}\}$, where $i \in \mathcal{K}_T$ is the maximum number with $i < j$ (for the smallest number $j \in \mathcal{K}_T$, we define $D(j) := 0$, but any other choice would fit the purpose).

It remains to count the number of instances in \mathcal{I} . We have $K + 1 \leq (|V| - 1)(|\mathcal{M}| - 1) + |\mathcal{M}| + 1$ choices for $L \in [c(H) + |\mathcal{J}|, c(H) + |\mathcal{J}| + K]$. For each pre-stay (M_{q_i}, v_i, σ_i) , there are $|\mathcal{M}|$ choices for M_{q_i} and $|V|$ choices for v_i . There is only one choice for σ_i : in the pre-stays for each machine M_q , σ increases from 1 to at most $|V||\mathcal{M}| + 2$ in steps of one by Definition 5.2(i). Thus, there are at most $|\mathcal{M}| \cdot (|V||\mathcal{M}| + 2)$ pre-stays in a pre-stay sequence T and hence, at most $(|V||\mathcal{M}|)^{|\mathcal{M}| \cdot (|V||\mathcal{M}| + 2)} \in 2^{O(|V||\mathcal{M}|^2 \log |V||\mathcal{M}|)}$ pre-stay sequences. Moreover, this implies that, for each pre-stay sequence T , one has $|\mathcal{K}_T| \leq |\mathcal{M}| \cdot (|V||\mathcal{M}| + 2)$. Thus, there are at most $(2|\mathcal{M}|)^{|\mathcal{M}| \cdot (|V||\mathcal{M}| + 2)} \in 2^{O(|V||\mathcal{M}|^2 \log |V||\mathcal{M}|)}$ length assignments $A: \mathcal{K}_T \rightarrow \{0, \dots, 2|\mathcal{M}| - 1\}$ and $(2|\mathcal{M}| + 1)^{|\mathcal{M}| \cdot (|V||\mathcal{M}| + 2)} \in 2^{O(|V||\mathcal{M}|^2 \log |V||\mathcal{M}|)}$ displacements $D: \mathcal{K}_T \rightarrow \{0, \dots, 2|\mathcal{M}|\}$. \square

Having Proposition 5.4, for proving Theorem 5.1, it remains to solve Problem 5.3 in $2^{O(|V||\mathcal{M}|^2 \log |V||\mathcal{M}|)} \cdot \text{poly}(|\mathcal{J}|)$ time since a shortest schedule for an instance I of ROUTING OPEN SHOP with unit processing times can be found by solving the instances $(I, (T, A, D), L) \in \mathcal{I}$ for increasing L .

5.2. Computing routes and completing the schedule

In this section, we provide the last missing ingredient for our fixed-parameter algorithm for ROUTING OPEN SHOP with unit processing times:

Proposition 5.8. Problem 5.3 can be solved in $2^{O(|V||\mathcal{M}|^2 \log |V||\mathcal{M}|)} \cdot \text{poly}(|\mathcal{J}|)$ time.

By Proposition 5.4, this proves Theorem 5.1. The key to our algorithm for Proposition 5.8 is the following lemma.

Lemma 5.9. Let $(I, (T, A, D), L)$ be an instance of Problem 5.3 that has a solution. Then, for arbitrary routes $(R_{M_q})_{M_q \in \mathcal{M}}$ of length L complying with (T, A, D) and satisfying $|\mathcal{T}_v^{R_{M_q}}| \geq |\mathcal{J}_v|$ for each non-critical vertex $v \in V$,

- (i) there is a partial schedule S with respect to $(R_{M_q})_{M_q \in \mathcal{M}}$ such that $S(J_i, M_q) \neq \perp$ if and only if $\mathcal{L}(J_i)$ is critical,
- (ii) any such partial schedule is completable with respect to routes $(R_{M_q})_{M_q \in \mathcal{M}}$.

Proof. We first show (ii). Let S be any partial schedule such that $S(J_i, M_q) \neq \perp$ if and only if $\mathcal{L}(J_i)$ is critical. We show that S is completable with respect to $(R_{M_q})_{M_q \in \mathcal{M}}$ in terms of Definition 3.2. For each critical vertex $v \in V$, each machine $M_q \in \mathcal{M}$, and each job $J_i \in \mathcal{J}_{M_q}^S \cap \mathcal{J}_v$, (3.1) of Definition 3.2 is trivially satisfied since there is no such job: $\mathcal{J}_{M_q}^S \cap \mathcal{J}_v = \emptyset$. For each non-critical vertex $v \in V$, each machine $M_q \in \mathcal{M}$, and each job $J_i \in \mathcal{J}_{M_q}^S \cap \mathcal{J}_v$, one has

$$|\mathcal{T}_v^{R_{M_q}} \setminus (\mathcal{T}_{J_i}^S \cup \mathcal{T}_{M_q}^S)| = |\mathcal{T}_v^{R_{M_q}}| \geq |\mathcal{J}_v| = \max\{|\mathcal{J}_v|, |\mathcal{M}|\},$$

thus satisfying (3.1) of Definition 3.2.

(i) Let S^* be a solution for $(I, (T, A, D), L)$, that is, S^* is a feasible schedule with respect to routes $(R_{M_q}^*)_{M_q \in \mathcal{M}}$ of length L complying with the pre-stay sequence $T = ((M_{q_k}, v_k, \sigma_k))_{k=1}^s$. We show how to construct a partial schedule S with respect to the given routes $(R_{M_q})_{M_q \in \mathcal{M}}$ such that $S(J_i, M_q) \neq \perp$ if and only if $\mathcal{L}(J_i)$ is critical. Let the routes for S^* be $(R_{M_q}^*)_{M_q \in \mathcal{M}}$, where $R_{M_q}^* = ((a_k^{q*}, w_k^{q*}, b_k^{q*}))_{k=1}^{t_q^*}$, and the given routes be $R_{M_q} = ((a_k^q, w_k^q, b_k^q))_{k=1}^{t_q}$ for each machine $M_q \in \mathcal{M}$. By Definition 5.2(ii) and Definition 1.1(iii), for each job $J_i \in \mathcal{J}$ and machine $M_q \in \mathcal{M}$ there is an index $P(J_i, M_q) := p$ of a pre-stay $(M_{q_p}, v_p, \sigma_p) = (M_q, \mathcal{L}(J_i), \sigma_p)$ on T such that

$$a_{\sigma_p}^{q*} \leq S^*(J_i, M_q) < b_{\sigma_p}^{q*}. \quad (5.1)$$

Since the routes $(R_{M_q}^*)_{M_q \in \mathcal{M}}$ and $(R_{M_q})_{M_q \in \mathcal{M}}$ comply with T , by Definition 5.2(ii), one has $t_q = t_q^*$ and, moreover, $w_k^q = w_k^{q*}$ for each machine $M_q \in \mathcal{M}$ and $1 \leq k \leq t_q$. For each job $J_i \in \mathcal{J}$ and machine $M_q \in \mathcal{M}$, we define

$$S(J_i, M_q) := \begin{cases} \perp & \text{if } \mathcal{L}(J_i) \text{ is not critical,} \\ S^*(J_i, M_q) - a_{\sigma_p}^{q*} + a_{\sigma_p}^q & \text{otherwise,} \end{cases}$$

where $p = P(J_i, M_q)$.

We show that S is indeed a partial schedule for the machine routes $(R_{M_q})_{M_q \in \mathcal{M}}$. For each job J_i in a critical vertex and each machine M_q , we first show that machine M_q stays in $\mathcal{L}(J_i)$ when processing job J_i . More precisely, for $p = P(J_i, M_q)$, we show $a_{\sigma_p}^q \leq S(J_i, M_q) < b_{\sigma_p}^q$ as follows. By adding $-a_{\sigma_p}^{q*} + a_{\sigma_p}^q$ to both sides of

$$a_{\sigma_p}^{q*} \leq S^*(J_i, M_q),$$

which holds since p is chosen so as to satisfy (5.1), one gets

$$a_{\sigma_p}^q \leq S^*(J_i, M_q) - a_{\sigma_p}^{q*} + a_{\sigma_p}^q = S(J_i, M_q).$$

Moreover, since both $R_{M_q}^*$ and R_{M_q} comply with the length assignment A , by Definition 5.2(iv), one has $b_{\sigma_k}^{q*} - a_{\sigma_k}^{q*} = A(k) = b_{\sigma_k}^q - a_{\sigma_k}^q$ for all pre-stays (M_{q_k}, v_k, σ_k) such that v_k is critical. Thus, by adding $-a_{\sigma_p}^{q*} + a_{\sigma_p}^q$ to both sides of

$$S^*(J_i, M_q) < b_{\sigma_p}^{q*},$$

which holds since p is chosen so as to satisfy (5.1), one gets

$$S(J_i, M_q) = S^*(J_i, M_q) - a_{\sigma_p}^{q*} + a_{\sigma_p}^q < b_{\sigma_p}^{q*} - a_{\sigma_p}^{q*} + a_{\sigma_p}^q = b_{\sigma_p}^q.$$

It remains to show that S^* processes no two jobs at the same time and that no two machines process one job at the same time. To this end, consider jobs $J_i, J_j \in \mathcal{J}$ in critical vertices and machines $M_q, M_r \in \mathcal{M}$. If either $J_i = J_j$ or $M_q = M_r$, then $S^*(J_i, M_q) \neq S^*(J_j, M_r)$. Thus, it is sufficient to show that $S^*(J_i, M_q) \neq S^*(J_j, M_r)$ implies $S(J_i, M_q) \neq S(J_j, M_r)$. To this end, let $p := P(J_i, M_q)$ and $\pi := P(J_j, M_r)$. Without loss of generality, assume that $p \leq \pi$. Then, by Definition 5.2(iii), one has $a_{\sigma_p}^q \leq a_{\sigma_\pi}^r$.

If $b_{\sigma_p}^q \leq a_{\sigma_\pi}^r$, then $S(J_i, M_q) \neq S(J_r, M_r)$ follows from $S(J_i, M_q) < b_{\sigma_p}^q \leq a_{\sigma_\pi}^r \leq S(J_j, M_r)$. Otherwise, since $b_{\sigma_p}^q - a_{\sigma_p}^q =$

$A(p) \leq 2|\mathcal{M}| - 1$ by Definition 5.2(iv), one has $a_{\sigma_\pi}^r - a_{\sigma_p}^q \leq b_{\sigma_p}^q - a_{\sigma_p}^q \leq 2|\mathcal{M}| - 1$. Thus, for $\mathcal{K}(p, \pi) := \{p < k \leq \pi \mid (M_{q_k}, v_k, \sigma_k) \text{ is a pre-stay of } T \text{ in a critical vertex}\}$, one has, by Definition 5.2(iii) and (v),

$$a_{\sigma_\pi}^r - a_{\sigma_p}^q = \sum_{k \in \mathcal{K}(p, \pi)} D(k) = a_{\sigma_\pi}^{r*} - a_{\sigma_p}^{q*} \quad (5.2)$$

since both tours $(R_{M_q})_{M_q \in \mathcal{M}}$ and $(R_{M_q}^*)_{M_q \in \mathcal{M}}$ comply with the displacement D . By adding $a_{\sigma_\pi}^r - a_{\sigma_p}^{q*}$ to both sides of

$$\begin{aligned} S(J_i, M_q) + a_{\sigma_\pi}^{q*} - a_{\sigma_p}^q &= S^*(J_i, M_q) \\ &\neq S^*(J_j, M_r) = S(J_j, M_r) + a_{\sigma_\pi}^{r*} - a_{\sigma_\pi}^r, \end{aligned}$$

which is true by the definition of S from S^* , one obtains

$$S(J_i, M_q) + a_{\sigma_\pi}^r - a_{\sigma_p}^q \neq S(J_r, M_r) + a_{\sigma_\pi}^{r*} - a_{\sigma_\pi}^{q*},$$

and, therefore, $S(J_i, M_q) \neq S(J_j, M_r)$ from (5.2). \square

Lemma 5.9 shows that, to solve Problem 5.3, it is sufficient to compute routes $(R_{M_q})_{M_q \in \mathcal{M}}$ of length L that comply with a given pre-schedule (T, A, D) and stay in each non-critical vertex v for at least $\lfloor \mathcal{J}_v \rfloor$ units of time. If no such routes are found, then the instance of Problem 5.3 has no schedule of length L since any feasible schedule has to spend at least $\lfloor \mathcal{J}_v \rfloor$ units of time in each vertex v . If such routes are found, then, by Lemma 5.9(ii), a feasible schedule for the non-critical vertices can be computed using the schedule completion Theorem 3.4. For critical vertices, we use the following brute force approach:

Lemma 5.10. Let $(I, (T, A, D), L)$ be an instance of Problem 5.3 and $(R_{M_q})_{M_q \in \mathcal{M}}$ be arbitrary routes complying with (T, A, D) .

If there is a partial schedule S for I that satisfies Lemma 5.9(i), then we can find it in $2^{O(|V||\mathcal{M}|^2 \log |\mathcal{M}|)} \cdot \text{poly}(|\mathcal{J}|)$ time.

Proof. Observe that, in total, there are at most $|V| \cdot |\mathcal{M}|$ jobs in critical vertices. Thus, we determine $S(J_i, M_q)$ for at most $|V| \cdot |\mathcal{M}|^2$ pairs $(J_i, M_q) \in \mathcal{J} \times \mathcal{M}$. By Lemma 5.6, each machine can process all of its jobs in a critical vertex staying there no longer than $2|\mathcal{M}| - 1$ units of time. Thus, for each of at most $|V| \cdot |\mathcal{M}|^2$ pairs (J_i, M_q) , we enumerate all possibilities of choosing $S(J_i, M_q)$ among the smallest $2|\mathcal{M}| - 1$ numbers in $\mathcal{T}_{\mathcal{L}(J_i)}^{R_{M_q}}$ and check each of them for feasibility. There are $(2|\mathcal{M}| - 1)^{|V| \cdot |\mathcal{M}|^2} \in 2^{O(|V||\mathcal{M}|^2 \log |\mathcal{M}|)}$ possibilities to do so. \square

Finally, we compute the routes required by Lemma 5.9 by testing the feasibility of an integer linear program with $O(|\mathcal{M}| \cdot (|V||\mathcal{M}| + 2))$ variables and constraints, which, by Lenstra's theorem, works in $2^{O(|V||\mathcal{M}|^2 \log |V||\mathcal{M}|)}$ time:

Theorem 5.11 (Lenstra (1983); also Kannan (1987)). A feasible solution to an integer linear program with p variables and m constraints is computable in $p^{O(p)} \cdot \text{poly}(m)$ time, if such a feasible solution exists.

Any feasible schedule has to stay in each non-critical vertex v for at least $\lfloor \mathcal{J}_v \rfloor$ time. Thus, the following lemma together with Lemmas 5.9 and 5.10 and Theorem 3.4 completes the proof of Proposition 5.8 and, hence, of Theorem 5.1.

Lemma 5.12. Let $(I, (T, A, D), L)$ be an instance of Problem 5.3.

In $2^{O(|V||\mathcal{M}|^2 \log |V||\mathcal{M}|)}$ time, one can compute routes $(R_{M_q})_{M_q \in \mathcal{M}}$ that have length L , comply with (T, A, D) , and satisfy $|\mathcal{T}_v^{R_{M_q}}| \geq |\mathcal{J}_v|$ for each non-critical vertex $v \in V$, if such routes exist.

Proof. Denote the given pre-stay sequence as $T = ((M_{q_i}, v_i, \sigma_i))_{i=1}^s$. For each machine $M_q \in \mathcal{M}$, let $t_q := \max\{\sigma \mid (M_q, w, \sigma) \text{ is a pre-stay on } T\}$. By Definition 5.2, $t_q \leq |V||\mathcal{M}| + 2$ for each machine $M_q \in \mathcal{M}$. We compute the routes $(R_{M_q})_{M_q \in \mathcal{M}}$, where $R_{M_q} := ((a_k^q, w_k^q, b_k^q))_{k=1}^{t_q}$, as follows. For each pre-stay (M_{q_i}, v_i, σ_i) on T , we let $w_{\sigma_i}^{q_i} := v_i$. If $w_1^q \neq v^*$ or $v^* \neq w_{t_q}^q$ for some machine M_q , where v^* is the depot, then there is no solution and we answer “no” accordingly. Otherwise, the a_k^q and b_k^q for each machine $M_q \in \mathcal{M}$ and $1 \leq k \leq t_q$ are at most $2|\mathcal{M}| \cdot (|V||\mathcal{M}| + 2)$ variables, which we determine using a feasible solution to an integer linear program. This, together with Theorem 5.11 directly yields the running time stated in Lemma 5.12.

Our linear program consists of the following constraints. We want each route to have length at most L , that is,

$$b_{t_q}^q \leq L \quad \text{for each } M_q \in \mathcal{M}.$$

A route must have sufficient travel time between stays, that is,

$$b_k^q + c(v_k^q, v_{k+1}^q) \leq a_{k+1}^q \quad \text{for each } M_q \in \mathcal{M} \\ \text{and } 1 \leq k \leq t_q - 1.$$

Stays should have non-negative length, that is,

$$a_k^q \leq b_k^q \quad \text{for each } M_q \in \mathcal{M} \\ \text{and } 1 \leq k \leq t_q.$$

Each machine should stay in $v \in V$ for at least $|\mathcal{J}_v|$ time, that is

$$\sum_{\substack{1 \leq k \leq t_q \\ w_k^q = v}} (b_k^q - a_k^q) \geq |\mathcal{J}_v| \quad \text{for each } M_q \in \mathcal{M} \text{ and } v \in V.$$

Stays must be ordered according to the pre-stay sequence T , that is

$$a_{\sigma_i}^{q_i} \leq a_{\sigma_j}^{q_j} \quad \text{for pre-stays } (M_{q_i}, v_i, \sigma_i) \\ \text{and } (M_{q_j}, v_j, \sigma_j) \text{ with } i \leq j.$$

Stays should adhere to the length assignment A , that is

$$b_{\sigma_i}^{q_i} - a_{\sigma_i}^{q_i} = A(i) \quad \text{for each pre-stay } (M_{q_i}, v_i, \sigma_i) \\ \text{such that } v_i \text{ is critical.}$$

Finally, the routes have to comply with the displacement D . To formulate the constraint, let $\mathcal{K} := \{i \leq s \mid v_i \text{ is critical}\}$ be the indices of pre-stays in critical vertices of T . For any two pre-stays (M_{q_i}, v_i, σ_i) and (M_{q_j}, v_j, σ_j) such that $i, j \in \mathcal{K}$ and $k \notin \mathcal{K}$ for all $k \in \{i+1, \dots, j-1\}$, we want that

$$a_{\sigma_j}^{q_j} \geq a_{\sigma_i}^{q_i} + D(j) \quad \text{if } D(j) = 2|\mathcal{M}|, \text{ and} \\ a_{\sigma_j}^{q_j} = a_{\sigma_i}^{q_i} + D(j) \quad \text{if } D(j) < 2|\mathcal{M}|. \quad \square$$

6. Conclusion

We have proved the schedule completion Theorem 3.4 and used it for a fixed-parameter algorithm for ROUTING OPEN SHOP with unit processing times. Precisely, we used it to prove upper bounds on various parameters of optimal schedules. This suggests that Theorem 3.4 will be likewise beneficial for approximation algorithms. Indeed, our Section 4 makes first steps into this direction.

A natural direction for future research is determining the parameterized complexity of ROUTING OPEN SHOP with unit processing times parameterized by the number $|V|$ of vertices. Even the question whether the problem is polynomial-time solvable for constant $|V|$ is open, yet we showed fixed-parameter tractability in the absence of critical vertices (Corollary 4.6). Finally, it would be desirable to find a fast polynomial-time algorithm for finding the coloring whose existence is witnessed by the theorem of Borodin et al. (1997) (Theorem 3.6).

References

- A. Allahverdi, C. Ng, T. Cheng, and M. Y. Kovalyov. A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, 187(3):985–1032, 2008. doi: 10.1016/j.ejor.2006.06.060.
- I. Averbakh, O. Berman, and I. Chernykh. A $\frac{6}{5}$ -approximation algorithm for the two-machine routing open-shop problem on a two-node network. *European Journal of Operational Research*, 166(1):3–24, 2005. doi: 10.1016/j.ejor.2003.06.050.
- I. Averbakh, O. Berman, and I. Chernykh. The routing open-shop problem on a network: Complexity and approximation. *European Journal of Operational Research*, 173(2):531–539, 2006. doi: 10.1016/j.ejor.2005.01.034.
- R. Bellman. Dynamic programming treatment of the Travelling Salesman Problem. *Journal of the ACM*, 9(1):61–63, 1962. doi: 10.1145/321105.321111.
- R. van Bevern and A. V. Pyatkin. Completing partial schedules for open shop with unit processing times and routing. In *Proceedings of the 11th International Computer Science Symposium in Russia (CSR'16)*, volume 9691 of *Lecture Notes in Computer Science*, pages 73–87. Springer, 2016. doi: 10.1007/978-3-319-34171-2_6.
- R. van Bevern, R. Niedermeier, M. Sorge, and M. Weller. Complexity of arc routing problems. In *Arc Routing: Problems, Methods, and Applications*, MOS-SIAM Series on Optimization. SIAM, 2014. doi: 10.1137/1.9781611973679.ch2.
- R. van Bevern, J. Chen, F. Hüffner, S. Kratsch, N. Talmon, and G. J. Woeginger. Approximability and parameterized complexity of multicover by c -intervals. *Information Processing Letters*, 115(10):744–749, 2015a. doi: 10.1016/j.ipl.2015.03.004.
- R. van Bevern, C. Komusiewicz, and M. Sorge. Approximation algorithms for mixed, windy, and capacitated arc routing problems. In *Proceedings of the 15th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems (ATMOS'15)*, volume 48 of *OpenAccess Series in Informatics (OASICS)*, pages 130–143. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2015b. doi: 10.4230/OASICS.ATMOS.2015.130.
- R. van Bevern, M. Mnich, R. Niedermeier, and M. Weller. Interval scheduling and colorful independent sets. *Journal of Scheduling*, 18:449–469, 2015c. doi: 10.1007/s10951-014-0398-5.
- R. van Bevern, R. Bredereck, L. Bulteau, C. Komusiewicz, N. Talmon, and G. J. Woeginger. Precedence-constrained scheduling problems parameterized by partial order width. In *Proceedings of the 2016 International Conference on Discrete Optimization and Operations Research (DOOR'16)*, 2016a. To appear, arxiv:1605.00901.
- R. van Bevern, R. Niedermeier, and O. Suchý. A parameterized complexity view on non-preemptively scheduling interval-constrained jobs: few machines, small looseness, and small slack. *Journal of Scheduling*, 2016b. doi: 10.1007/s10951-016-0478-9. In press.
- H. L. Bodlaender and M. R. Fellows. W[2]-hardness of precedence constrained k -processor scheduling. *Operations Research Letters*, 18(2):93–97, 1995. doi: 10.1016/0167-6377(95)00031-9.

- O. V. Borodin, A. V. Kostochka, and D. R. Woodall. List edge and list total colourings of multigraphs. *Journal of Combinatorial Theory, Series B*, 71(2): 184–204, 1997. doi: 10.1006/jctb.1997.1780.
- H.-J. Böckenhauer, J. Hromkovič, J. Kneis, and J. Kupke. The parameterized approximability of TSP with deadlines. *Theory of Computing Systems*, 41(3): 431–444, 2007. doi: 10.1007/s00224-007-1347-x.
- I. Chernykh, A. V. Kononov, and S. Sevastyanov. Efficient approximation algorithms for the routing open shop problem. *Computers & Operations Research*, 40(3):841–847, 2013. doi: 10.1016/j.cor.2012.01.006.
- M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer, 2015. doi: 10.1007/978-3-319-21275-3.
- F. Dorn, H. Moser, R. Niedermeier, and M. Weller. Efficient algorithms for Eulerian Extension and Rural Postman. *SIAM Journal on Discrete Mathematics*, 27(1):75–94, 2013. doi: 10.1137/110834810.
- R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013. doi: 10.1007/978-1-4471-5559-1.
- M. R. Fellows and C. McCartin. On the parametric complexity of schedules to minimize tardy tasks. *Theoretical Computer Science*, 298(2):317–324, 2003. doi: 10.1016/S0304-3975(02)00811-3.
- R. W. Floyd. Algorithm 97: Shortest path. *Communications of the ACM*, 5(6): 345, 1962. doi: 10.1145/367766.368168.
- J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006. doi: 10.1007/3-540-29953-X.
- F. Galvin. The list chromatic index of a bipartite multigraph. *Journal of Combinatorial Theory, Series B*, 63(1):153–158, 1995. doi: 10.1006/jctb.1995.1011.
- T. Gonzalez and S. Sahni. Open shop scheduling to minimize finish time. *Journal of the ACM*, 23(4):665–679, 1976. doi: 10.1145/321978.321985.
- G. Gutin, G. Muciaccia, and A. Yeo. Parameterized complexity of k -Chinese Postman Problem. *Theoretical Computer Science*, 513:124–128, 2013. doi: 10.1016/j.tcs.2013.10.012.
- G. Gutin, M. Jones, and B. Sheng. Parameterized complexity of the k -Arc Chinese Postman Problem. In *Proceedings of the 22rd Annual European Symposium on Algorithm (ESA'14)*, volume 8737 of *Lecture Notes in Computer Science*, pages 530–541. Springer, 2014a. doi: 10.1007/978-3-662-44777-2_44.
- G. Gutin, M. Wahlström, and A. Yeo. Rural Postman parameterized by the number of components of required edges. *Journal of Computer and System Sciences*, 2014b. To appear, arXiv:1308.2599v4.
- G. Gutin, M. Jones, and M. Wahlström. Structural parameterizations of the Mixed Chinese Postman Problem. In *Proceedings of the 23rd Annual European Symposium on Algorithm (ESA'15)*, volume 9294 of *Lecture Notes in Computer Science*, pages 668–679. Springer, 2015. doi: 10.1007/978-3-662-48350-3_56.
- M. M. Halldórsson and R. K. Karlsson. Strip graphs: Recognition and scheduling. In *Proceedings of the 32nd International Workshop on Graph-Theoretic Concepts in Computer Science (WG'06)*, volume 4271 of *Lecture Notes in Computer Science*, pages 137–146. Springer, 2006. doi: 10.1007/11917496_13.
- M. Held and R. M. Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, 10(1):196–210, 1962. doi: 10.1137/0110015.
- D. Hermelin, J.-M. Kubitza, D. Shabtay, N. Talmon, and G. Woeginger. Scheduling two competing agents when one agent has significantly fewer jobs. In *Proceedings of the 10th International Symposium on Parameterized and Exact Computation (IPEC'15)*, volume 43 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 55–65. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2015. doi: 10.4230/LIPIcs.IPEC.2015.55.
- R. Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987. doi: 10.1287/moor.12.3.415.
- P. N. Klein and D. Marx. A subexponential parameterized algorithm for Subset TSP on planar graphs. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'14)*, pages 1812–1830. Society for Industrial and Applied Mathematics, 2014. doi: 10.1137/1.9781611973402.131.
- A. Kononov. $O(\log n)$ -approximation for the routing open shop problem. *RAIRO Operations Research*, 49(2):383–391, 2015. doi: 10.1051/ro/2014051.
- H. W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983. doi: 10.1287/moor.8.4.538.
- M. Mnich and A. Wiese. Scheduling and fixed-parameter tractability. *Mathematical Programming*, 154(1-2):533–562, 2015. doi: 10.1007/s10107-014-0830-9.
- R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006. doi: 10.1093/acprof:oso/9780198566076.001.0001.
- A. V. Pyatkin and I. D. Chernykh. The Open Shop problem with routing in a two-node network and allowed preemption (In Russian). *Diskretnyj Analiz i Issledovaniye Operatsij*, 19(3):65–78, 2012. English translation in *Journal of Applied and Industrial Mathematics*, 6(3):346–354.
- A. I. Serdyukov. On some extremal by-passes in graphs (In Russian). *Upravlyayemye sistemy*, 17:76–79, 1978. English abstract in zbMATH 0475.90080.
- M. Sorge, R. van Bevern, R. Niedermeier, and M. Weller. From few components to an Eulerian graph by adding arcs. In *Proceedings of the 37th International Workshop on Graph-Theoretic Concepts in Computer Science (WG'11)*, pages 307–318. Springer, 2011. doi: 10.1007/978-3-642-25870-1_28.
- M. Sorge, R. van Bevern, R. Niedermeier, and M. Weller. A new view on Rural Postman based on Eulerian Extension and Matching. *Journal of Discrete Algorithms*, 16:12–33, 2012. doi: 10.1016/j.jda.2012.04.007.
- W. Yu, Z. Liu, L. Wang, and T. Fan. Routing open shop and flow shop scheduling problems. *European Journal of Operational Research*, 213(1):24–36, 2011. doi: 10.1016/j.ejor.2011.02.028.
- X. Zhu and W. E. Wilhelm. Scheduling and lot sizing with sequence-dependent setup: A literature review. *IIE Transactions*, 38(11):987–1007, 2006. doi: 10.1080/07408170600559706.